Data Mining Fundamentals

Chapter 10. Cluster Analysis: Basic Concepts and Methods

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Density- and Grid-Based Methods
- Evaluation of Clustering (Coverage will be based on the available time)
- Summary

What Is Cluster Analysis?

- **What is a cluster?**
  - A cluster is a collection of data objects which are
    - Similar (or related) to one another within the same group (i.e., cluster)
    - Dissimilar (or unrelated) to the objects in other groups (i.e., clusters)
- **Cluster analysis** (or ***clustering*, *data segmentation*, …**)
  - Given a set of data points, partition them into a set of groups (i.e., clusters) which are as similar as possible
- Cluster analysis is **unsupervised learning** (i.e., no predefined classes)
  - This contrasts with *classification* (i.e., *supervised learning*)
- Typical ways to use/apply cluster analysis
  - As a stand-alone tool to get insight into data distribution, or
  - As a preprocessing (or intermediate) step for other algorithms

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Density- and Grid-Based Methods
- Evaluation of Clustering
- Summary

Partitioning-Based Clustering Methods

- Basic Concepts of Partitioning Algorithms
- The K-Means Clustering Method
- Initialization of K-Means Clustering
- The K-Medoids Clustering Method
- The K-Medians and K-Modes Clustering Methods
- The Kernel K-Means Clustering Method

Partitioning Algorithms: Basic Concepts

- Partitioning method: Discovering the groupings in the data by optimizing a specific objective function and iteratively improving the quality of partitions
- *K*-partitioning method: Partitioning a dataset ***D*** of ***n*** objects into a set of ***K*** clusters so that an objective function is optimized (e.g., the sum of squared distances is minimized, where c_k is the centroid or medoid of cluster C_k)
  - A typical objective function: **Sum of Squared Errors (SSE)**

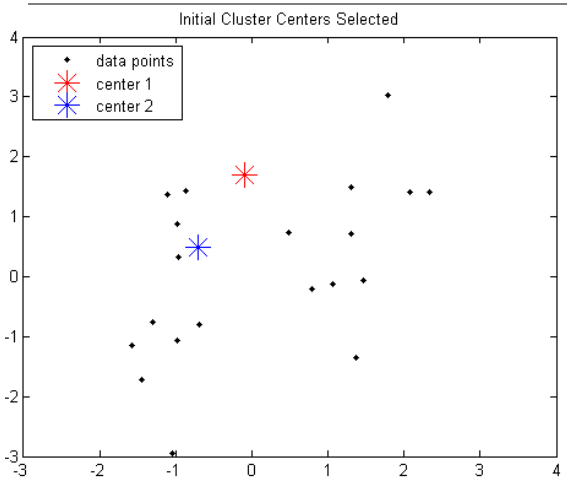
$$SSE(C) = \sum_{k=1}^{K} \sum_{x_{i \in C_k}} || x_i - c_k ||^2$$

- Problem definition: Given *K*, find a partition of *K clusters* that optimizes the chosen partitioning criterion
  - Global optimal: Needs to exhaustively enumerate all partitions
  - Heuristic methods (i.e., greedy algorithms): *K-Means*, *K-Medians*, *K-Medoids*, etc.

The *K-Means* Clustering Method

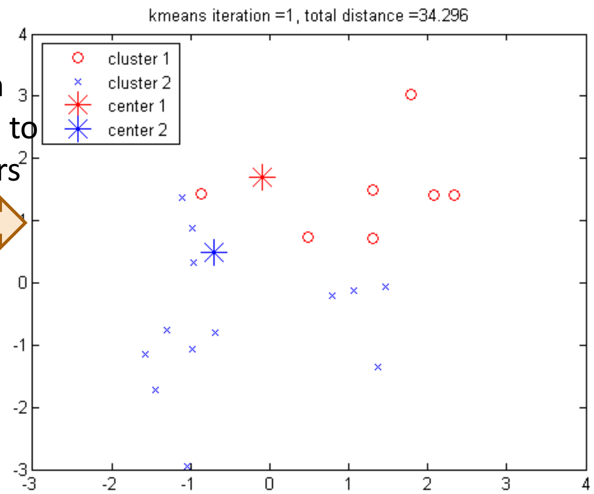
- *K-Means* (MacQueen'67, Lloyd'57/'82)
  - Each cluster is represented by the center of the cluster
- Given K, the number of clusters, the *K-Means* clustering algorithm is outlined as follows
  - Select *K* points as initial centroids
  - **Repeat**
    - Form *K* clusters by assigning each point to its closest centroid
    - Re-compute the centroids (i.e., *mean point*) of each cluster
  - **Until** convergence criterion is satisfied
- Different kinds of measures can be used
  - Manhattan distance (L_1 norm), Euclidean distance (L_2 norm), Cosine similarity

Example: *K-Means* Clustering

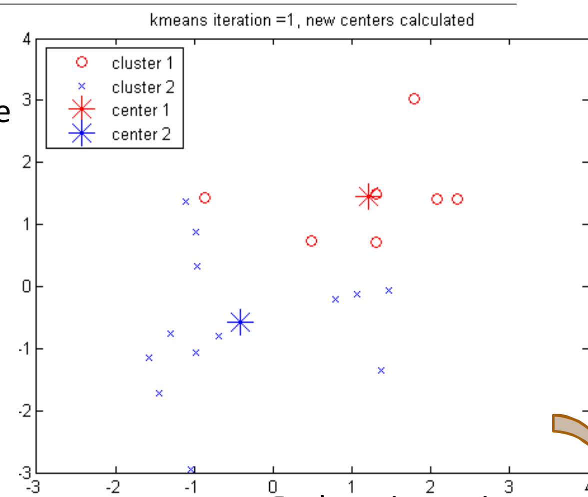


The original data points &
randomly select K = 2 centroids

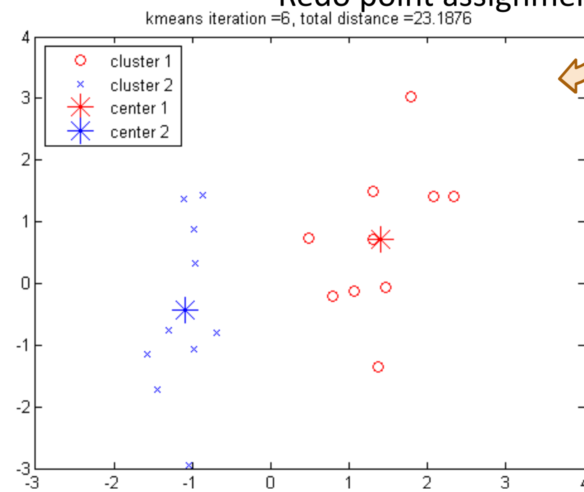
Assign points to clusters



Recompute cluster centers



Redo point assignment



Execution of the K-Means Clustering Algorithm

Select K points as initial centroids

Repeat

- Form K clusters by assigning each point to its closest centroid
- Re-compute the centroids (i.e., *mean point*) of each cluster

Until convergence criterion is satisfied

Discussion on the *K-Means* Method

- **Efficiency**: $O(tKn)$ where n: # of objects, K: # of clusters, and t: # of iterations
  - Normally, K, $t << n$; thus, an efficient method
- K-means clustering often ***terminates at a local optimal***
  - Initialization can be important to find high-quality clusters
- **Need to specify *K*,** the *number* of clusters, in advance
  - There are ways to automatically determine the "*best*" K
  - In practice, one often runs a range of values and selected the "*best*" K value
- **Sensitive to noisy data and *outliers***
  - Variations: Using K-medians, K-medoids, etc.
- K-means is applicable only to objects in a continuous n-dimensional space
  - Using the K-modes for ***categorical data***
- Not suitable to discover clusters with ***non-convex shapes***
  - Using density-based clustering, kernel *K*-means, etc.

Variations of *K-Means*

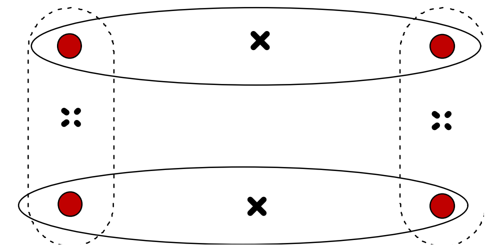
- There are many variants of the *K-Means* method, varying in different aspects
  - Choosing better initial centroid estimates
    - *K-means++, Intelligent K-Means, Genetic K-Means*

      To be discussed in this lecture
  - Choosing different representative prototypes for the clusters
    - *K-Medoids, K-Medians, K-Modes*

      To be discussed in this lecture
  - Applying feature transformation techniques
    - *Weighted K-Means, Kernel K-Means*

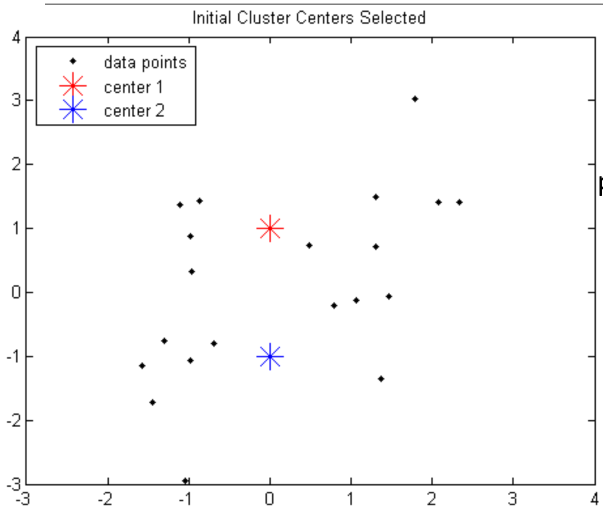
      To be discussed in this lecture

Initialization of K-Means



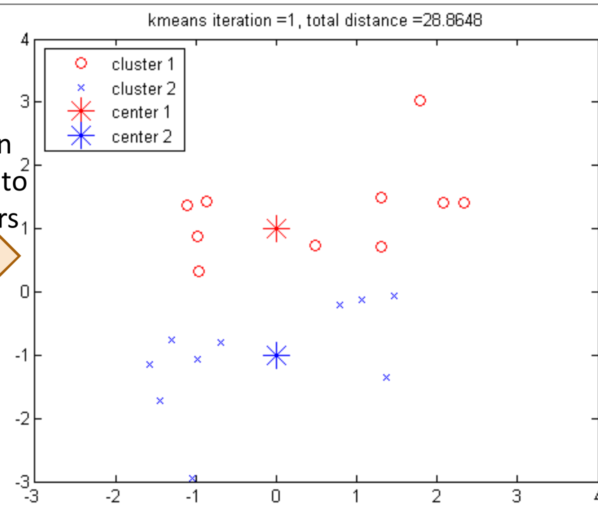
- Different initializations may generate rather different clustering results (some could be far from optimal)
- Original proposal (MacQueen'67): Select *K* seeds randomly
  - Need to run the algorithm multiple times using different seeds
- There are many methods proposed for better initialization of *k* seeds
  - ***K-Means++*** (Arthur & Vassilvitskii'07):
    - The first centroid is selected at random
    - The next centroid selected is the one that is farthest from the currently selected (selection is based on a weighted probability score)
    - The selection continues until *K* centroids are obtained

Example: Poor Initialization May Lead to Poor Clustering

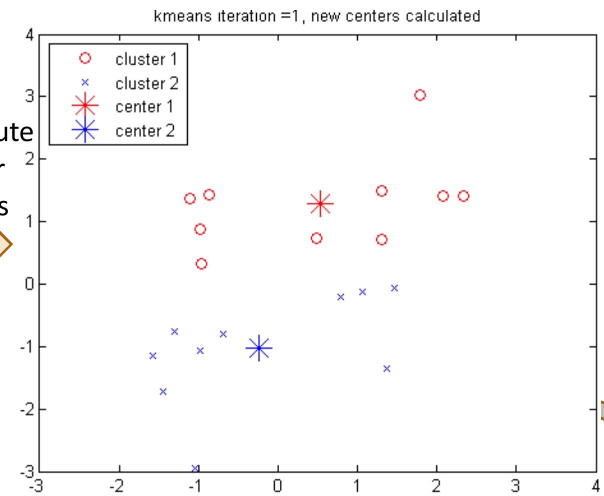


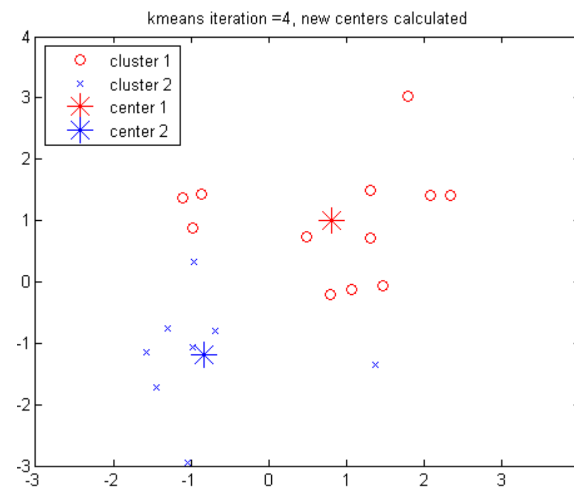
Another random selection of k centroids for the same data points

Assign points to clusters



Recompute cluster centers



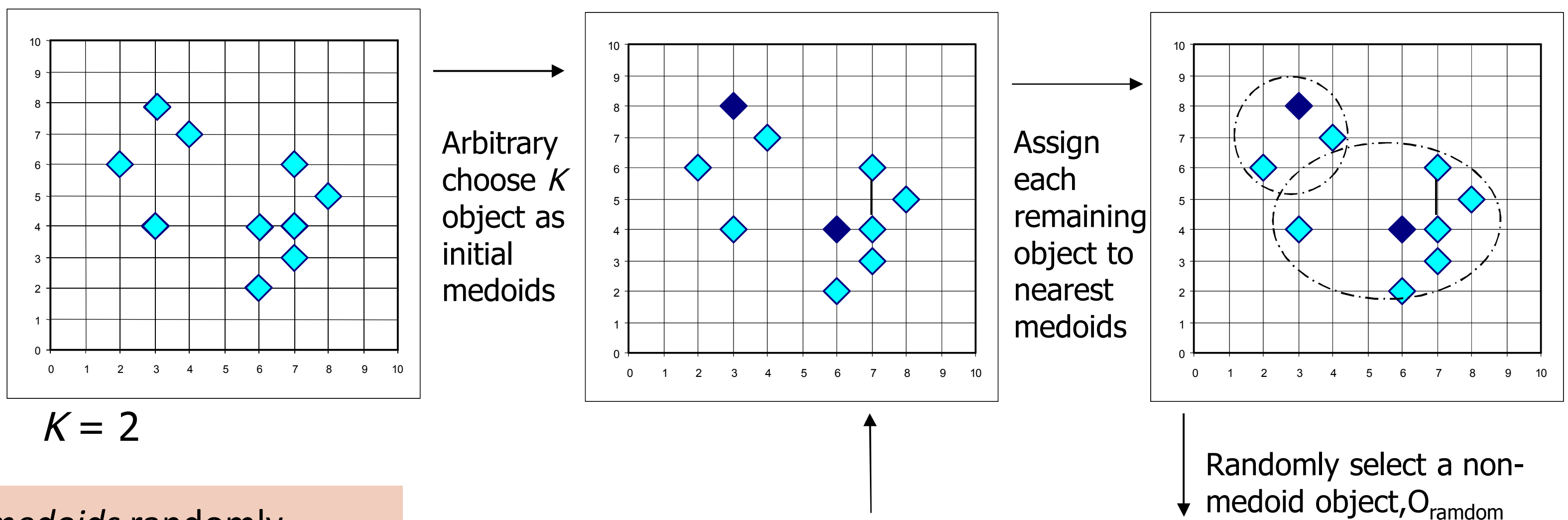


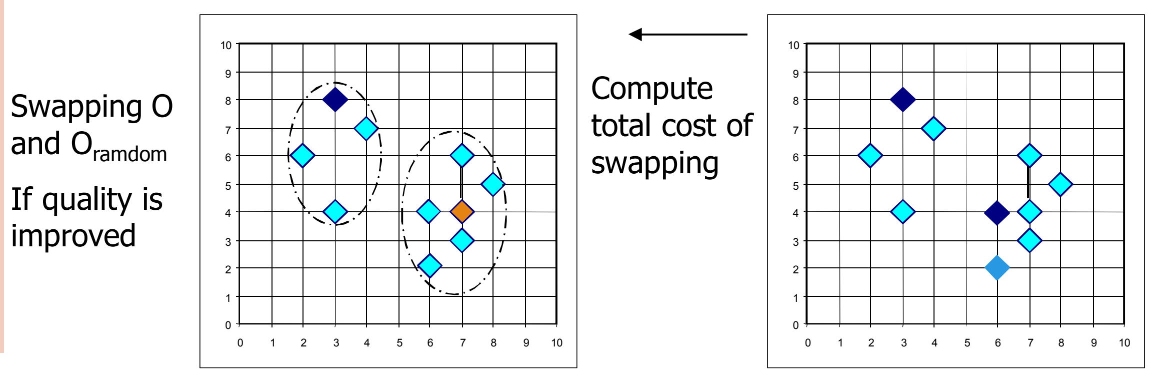
- Rerun of the *K-Means* using another random *K* seeds
- This run of *K*-Means generates a poor quality clustering

Handling Outliers: From *K-Means* to *K-Medoids*

- The *K-Means* algorithm is sensitive to outliers!—since an object with an extremely large value may substantially distort the distribution of the data
- *K-Medoids*: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster
- The *K-Medoids* clustering algorithm:
  - Select *K* points as the initial representative objects (i.e., as initial *K medoids*)
  - **Repeat**
    - Assigning each point to the cluster with the closest medoid
    - Randomly select a non-representative object o_i
    - Compute the total cost S of swapping the medoid m with o_i
    - If $S < 0$, then swap m with o_i to form the new set of medoids
  - **Until** convergence criterion is satisfied

PAM: A Typical *K-Medoids* Algorithm





Select initial *K medoids* randomly

Repeat

Object re-assignment

Swap medoid *m* with o_i if it improves the clustering quality

Until convergence criterion is satisfied

Discussion on *K-Medoids* Clustering

- *K-Medoids* Clustering: Find *representative* objects (medoids) in clusters
- *PAM* (Partitioning Around Medoids: Kaufmann & Rousseeuw 1987)
  - Starts from an initial set of medoids, and
  - Iteratively replaces one of the medoids by one of the non-medoids if it improves the total sum of the squared errors (SSE) of the resulting clustering
  - *PAM* works effectively for small data sets but does not scale well for large data sets (due to the computational complexity)
  - Computational complexity: PAM: $O(K(n - K)^2)$ (quite expensive!)
- Efficiency improvements on PAM
  - *CLARA* (Kaufmann & Rousseeuw, 1990):
    - PAM on samples; $O(Ks^2 + K(n - K))$, s is the sample size
  - *CLARANS* (Ng & Han, 1994): Randomized re-sampling, ensuring efficiency + quality

K-Medians: Handling Outliers by Computing Medians

- Medians are less sensitive to outliers than means
  - Think of the median salary vs. mean salary of a large firm when adding a few top executives!
- ***K-Medians***: Instead of taking the **mean** value of the object in a cluster as a reference point, **medians** are used (L_1-norm as the distance measure)
- The criterion function for the *K-Medians* algorithm:

$$S = \sum_{k=1}^{K} \sum_{x_{i \in C_k}} | x_{ij} - med_{kj} |$$

- The *K-Medians* clustering algorithm:
  - Select *K* points as the initial representative objects (i.e., as initial *K medians*)
  - **Repeat**
    - Assign every point to its nearest median
    - Re-compute the median using the median of each individual feature
  - **Until** convergence criterion is satisfied

K-Modes: Clustering Categorical Data

- *K-Means* cannot handle non-numerical (categorical) data
  - Mapping categorical value to 1/0 cannot generate quality clusters for high-dimensional data
- ***K-Modes***: An extension to *K-Means* by replacing means of clusters with ***modes***
- Dissimilarity measure between object X and the center of a cluster Z
  - $\Phi(x_j, z_j) = 1 - n_j^r/n_l$ when $x_j = z_j$; 1 when $x_j \neq z_j$
    - where z_j is the categorical value of attribute j in Z_l, n_l is the number of objects in cluster *l*, and n_j^r is the number of objects whose attribute value is r
- This dissimilarity measure (distance function) is **frequency-based**
- Algorithm is still based on iterative *object cluster assignment* and *centroid update*
- A ***fuzzy K-Modes*** method is proposed to calculate a ***fuzzy cluster membership value*** for each object to each cluster
- A mixture of categorical and numerical data: Using a ***K-Prototype*** method